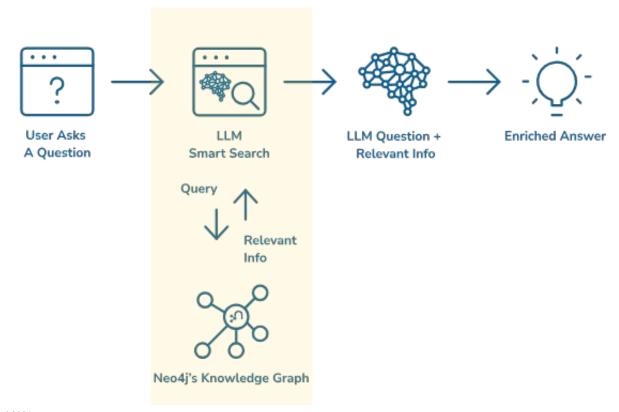
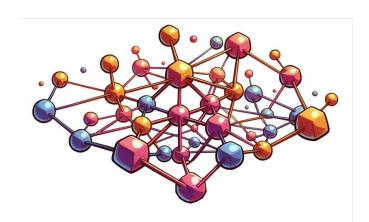
Going Meta #24: KG+LLMs: Ontology driven RAG patterns

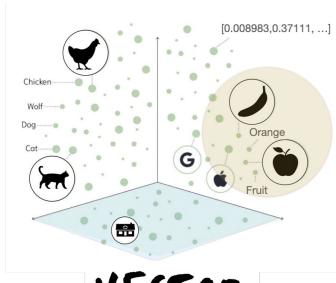
Recap from last episode... RAG



Advanced RAG patterns combine vector & graph...



GRAPH



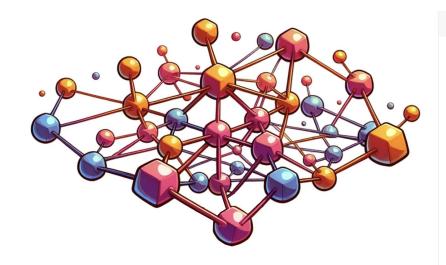




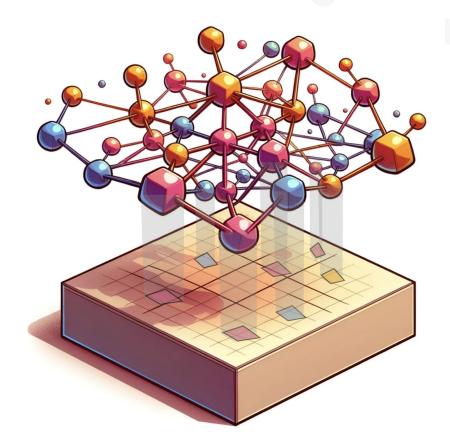


A **Knowledge Graph** captures key enterprise knowledge in the form of **entities and relationships between them**.

Some nodes in the graph have properties with NL text

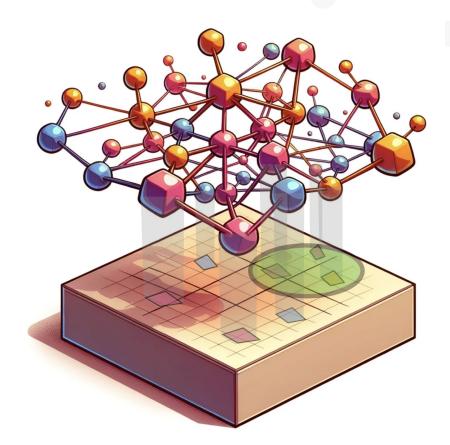


These property values get **embedded** (transformed into a numeric representation) and added to a vector index to enable **vector-based semantic search**.



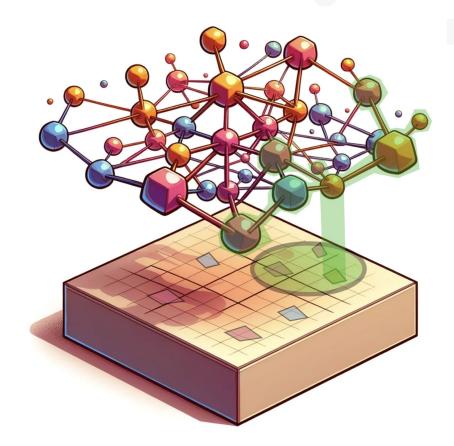


A semantic search on the vector index returns the k approximate nearest neighbours to the search concept (word, question, image, etc)



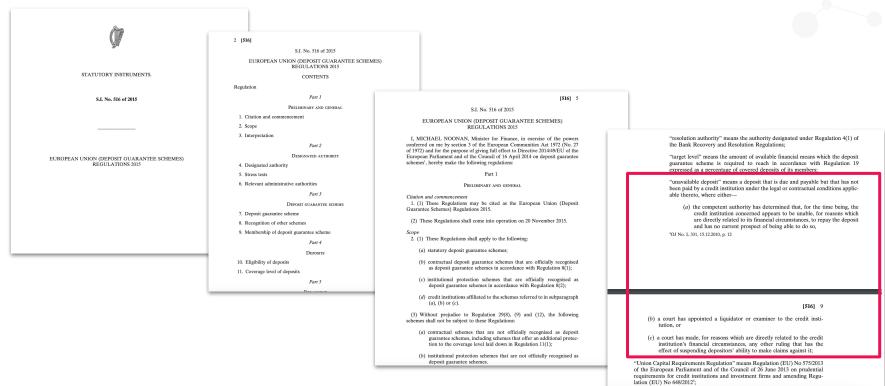


Each result from the vector search is "dereferenced" to get the corresponding node in the graph and a subsequent graph exploration finds semantically related elements that enrich and augment the final search result.





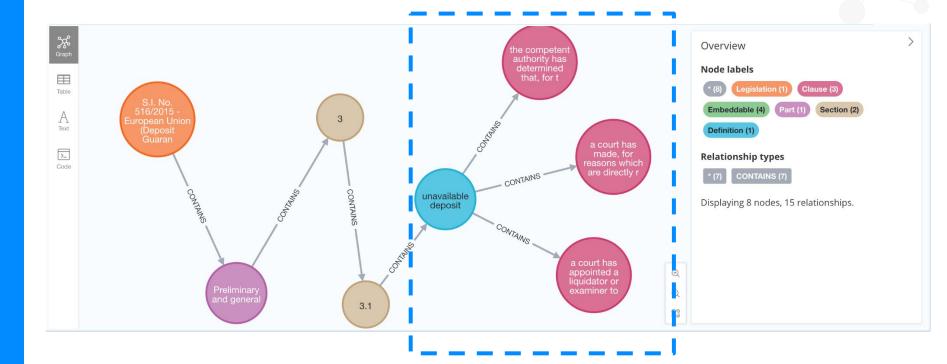
Example1: Q&A on document with rich internal structure



https://www.irishstatutebook.ie/eli/2015/si/516/made/en/pdf



The document as a graph





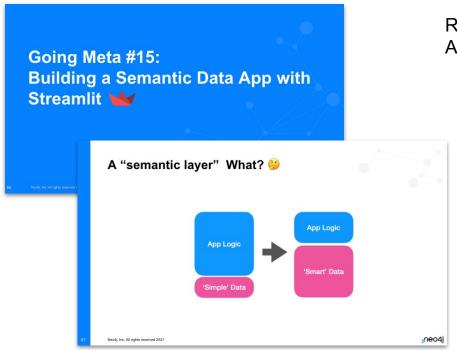
Injecting graph post-processing (powerful 6 but rigid 2)



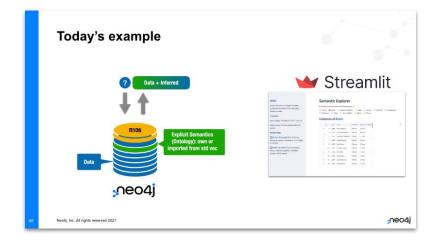


```
contextualize query =
...cypher magic...
11 11 11
contextualized vectorstore = Neo4jVector.from existing index(
   OpenAIEmbeddings(),
   url=url,
   username=username,
   password=password,
   index name="legislation",
   retrieval query=contextualize query,
```

Let's go meta!



Remember episode 15?
An "ontology driven" streamlit app





Let's see some code, ok?

